The complexity of FREE FLOOD IT on $2 \times n$ boards

Kitty Meeks and Alexander Scott
Mathematical Institute, University of Oxford, 24-29 St Giles, Oxford OX1 3LB, UK
{meeks,scott}@maths.ox.ac.uk

December 14, 2012

Abstract

We consider the complexity of problems related to the combinatorial game Free-Flood-It, in which players aim to make a coloured graph monochromatic with the minimum possible number of flooding operations. Our main result is that computing the length of an optimal sequence is fixed parameter tractable (with the number of colours as a parameter) when restricted to rectangular $2 \times n$ boards. We also show that, when the number of colours is unbounded, the problem remains NP-hard on such boards. These results resolve a question of Clifford, Jalsenius, Montanaro and Sach.

1 Introduction

In this paper we consider the complexity of problems related to the one-player combinatorial game Flood-It, introduced by Arthur, Clifford, Jalsenius, Montanaro and Sach in [5]. The original game is played on a board consisting of an $n \times n$ grid of coloured squares, each square given a colour from some fixed colour-set, but we can more generally regard the game as being played on a vertex-coloured graph. A move then consists of picking a vertex v and a colour d, and giving all vertices in the same monochromatic component as v colour d. The goal is to make the entire graph monochromatic with as few such moves as possible.

When the game is played on a planar graph, it can be regarded as modelling repeated use of the flood-fill tool in Microsoft Paint. Implementations of the game, played on a square grid, are widely available online, and include a flash game [1] as well as popular smartphone apps [2, 3]. There also exist implementations using a hexagonal grid: Mad Virus [4] is the same one-player game described above, while the Honey Bee Game [6] is a two player variant, and has been studied by Fleischer and Woeginger [9]. All these implementations are based on the "fixed" version of the game, where all moves must be played at the same fixed vertex (usually the vertex corresponding to the top left square when the board is an $n \times n$ grid).

For any coloured graph, we define the following problems.

- Free-Flood-It is the problem of determining the minimum number of moves required to flood the graph, if we are allowed to make moves anywhere in the graph.
- Fixed-Flood-It is the same problem when all moves must be played at a single specified vertex.[1]
- c-Free-Flood-It and c-Fixed-Flood-It respectively are the variants of Free-Flood-It and Fixed-Flood-It in which only colours from some fixed set of size c are used.

Note that we can trivially flood an n-vertex graph with $n - 1$ moves, and that if c colours are present in the initial colouring we require at least $c - 1$ moves.

These problems are known to be computationally difficult in many situations. In [5], Arthur, Clifford, Jalsenius, Montanaro and Sach proved that c-Free-Flood-It is NP-hard in the case of an $n \times n$ grid, for every $c \geq 3$, and that this result also holds for the fixed variant. Lagoutte, Noual and Thierry [12, 13] showed that the same result holds when the game is played instead on a hexagonal grid, as in Mad Virus or a one-player version of the Honey Bee Game. Fleischer and Woeginger [9] proved that c-Fixed Flood It remains NP-hard when restricted to trees, for every $c \geq 4$,[2] and Fukui, Nakanishi, Uehara, Uno and Uno [10] demonstrated that this result can be extended to show the hardness c-Free Flood It under the same conditions.

A few positive results are known, however. 2-Free-Flood-It is solvable in polynomial time on arbitrary graphs, a result shown independently by Clifford et. al. [7], Lagoutte [12] and Meeks and Scott [14]. It is also known that Fixed-Flood-It and Free-Flood-It are solvable in polynomial time on paths [7, 14, 10] and cycles [10], and more generally on any graph with

[1]Fixed Flood It is often referred to as simply Flood-It, but we use the longer name to avoid confusion with the free version.

[2]Note that this proof does in fact require four colours, not three as stated in a previous version of [9].

only a polynomial number of connected subgraphs [15, 16]. Meeks and Scott also show that the number of moves required to create a monochromatic component containing an arbitrary, bounded-size subset of the vertices can be computed in polynomial time, even when the number of colours is unbounded [16, 15].

A major focus of previous research has been the restriction of the game to rectangular boards of fixed height. Although an additive approximation for c-Free-Flood-It can be computed in polynomial time [14], solving either c-Free-Flood-It or c-Fixed-Flood-It exactly remains NP-hard on $3 \times n$ boards, whenever $c \geq 4$ [14]. However, Clifford et. al. [7] give a linear time algorithm for Fixed-Flood-It on $2 \times n$ boards. They also raise the question of the complexity of the free variant in this setting.

Here we address this remaining case of (c-)Free-Flood-It restricted to $2 \times n$ boards, which turn out to be a particularly interesting class of graphs on which to analyse the game. The majority of the paper describes an algorithm to demonstrate that c-Free-Flood-It, restricted to $2 \times n$ boards, is fixed parameter tractable with parameter c. To do this we exploit some general results from [16] about the relationship between the number of moves required to flood a graph and its spanning trees.

On the other hand, we also show that Free-Flood-It remains NP-hard in this setting. This is a somewhat surprising result, as it gives the first example of a class of graphs on which the complexity of Fixed-Flood-It and Free-Flood-It has been shown to be different.

The rest of the paper is organised as follows. We begin with notation and definitions in Section 2, before giving our algorithm for c-Free-Flood-It in Section 3. Finally, in Section 4, we show that the problem remains NP-hard when the number of colours used is unbounded.

2 Notation and definitions

Although the original Flood-It game is played on a square grid, and our main results here concern the game restricted to a rectangular grid, it is convenient to consider the generalisation of the game to an arbitrary graph $G = (V, E)$, equipped with an initial colouring ω using colours from the *colour-set* C. Then each move $m = (v, d)$ consists of choosing some vertex $v \in V$ and a colour $d \in C$, and assigning colour d to all vertices in the same monochromatic component as v. The goal is to give every vertex in G the same colour, using as few moves as possible.

Given any connected graph G, equipped with a colouring ω (not necessarily proper), we define $m(G, \omega, d)$ to be the minimum number of moves required in the free variant to give all its vertices colour d, and $m(G,\omega)$ to be $\min_{d\in C} m(G,\omega,d)$. If S is a sequence of moves played on a graph G with initial colouring ω, we denote by $S(\omega,G)$ the new colouring obtained by playing S in G. Note that, if the initial colouring ω of G is not proper, we may obtain an equivalent coloured graph G' (with colouring ω') by contracting monochromatic components of G with respect to ω.

Let A be any subset of V. We denote by $\mathrm{col}(A,\omega)$ the set of colours assigned to vertices of A by ω. We say a move $m=(v,d)$ is *played in* A if $v\in A$, and that A is *linked* if it is contained in a single monochromatic component. Subsets $A,B\subseteq V$ are *adjacent* if there exists $ab\in E$ with $a\in A$ and $b\in B$.

When we consider the game played on a rectangular board B, we are effectively playing the game in a corresponding coloured graph G, obtained from the planar dual of B (in which there is one vertex corresponding to each square of B, and vertices are adjacent if they correspond to squares which are either horizontally or vertically adjacent in B) by giving each vertex the colour of the corresponding square in B. We identify areas of B with the corresponding subgraphs of G, and may refer to them interchangeably.

We define a *border* of B to be a union of edges of squares on the original board B that forms a path from the top edge of the board to the bottom (but not including any edges that form the top or bottom edge of the board). Thus, a border in B corresponds to an edge-cut in the corresponding graph. Observe that a border is uniquely defined by the points at which it meets the top and bottom of the board, so there are $(n+1)^2$ borders in total. We denote by b_L and b_R the borders corresponding to the left-hand and right-hand edges of the board respectively. Given two borders b_1 and b_2, we write $b_1\le b_2$ if and only if b_1 meets both the top and bottom of the board to the left of (or at the same point as) b_2, and write $b_1<b_2$ if $b_1\le b_2$ and $b_1\ne b_2$. Note that if $b_1\le b_2$ then b_1 lies entirely to the left of b_2 (the two borders may meet but never cross); this is a special property of $2\times n$ boards and does not hold for $k\times n$ boards for $k\ge 3$.

If G is the graph corresponding to the $2\times n$ board B, we say that a vertex (or subgraph) is *incident* with a border b if the vertex (or some vertex in the subgraph) corresponds to a square on B whose edge forms part of b. If $b_1<b_2$ are borders, we denote the subgraph induced by vertices lying between b_1 and b_2 by $B[b_1,b_2]$, and we say $B[b_1,b_2]$ is a *section* if it is connected.

Finally, given any tree T, we denote by $\mathrm{trunk}(T)$ the subtree obtained by deleting all leaves of T, and given any $x,y\in V(T)$ we set $P(T,x,y)$ to be the unique path from x to y in T.

3 c-FREE FLOOD IT on $2 \times n$ boards

In this section, we give an algorithm to solve c-Free-Flood-It on $2 \times n$ boards. More specifically, we prove the following result, which shows that c-Free-Flood-It, restricted to $2 \times n$ boards, is fixed parameter tractable, parameterised by c. This answers an open question of Clifford, Jalsenius, Montanaro and Sach [7].

Theorem 3.1. *When restricted to $2 \times n$ boards, c-Free-Flood-It can be solved in time $O(n^{11} \cdot 2^c)$.*

We begin with some background and auxiliary results in Section 3.1, and then describe the algorithm in Section 3.2.

3.1 Background and auxiliary results

Before describing our algorithm in the next section, we need a number of results which will be used to prove its correctness. We begin with some previous results from [16]. Meeks and Scott prove that it suffices to consider spanning trees in order to determine the minimum number of moves required to flood a graph. For any connected graph G, let $\mathcal{T}(G)$ denote the set of all spanning trees of G.

Theorem 3.2. *Let G be a connected graph with colouring ω from colour-set C. Then, for any $d \in C$,*

$$m(G, \omega, d) = \min_{T \in \mathcal{T}(G)} m(T, \omega, d).$$

For any $d \in C$, we say that T is a *d-minimal* spanning tree for G if $m(T, \omega, d) = m(G, \omega, d)$.

In the remainder of this section, we prove that in the special case in which G corresponds to a $2 \times n$ board, there is always a d-minimal spanning tree T such that trunk(T) is a path.

In doing so, and in proving the correctness of our algorithm in the next section, we make use of a corollary of Theorem 3.2, again proved in [16], which shows that the number of moves required to flood a graph is bounded above by the sum of the numbers of moves required to flood connected subgraphs which cover the vertex-set.

Corollary 3.3. *Let G be a connected graph, with colouring ω from colour-set C, and let A and B be subsets of $V(G)$ such that $V(G) = A \cup B$ and $G[A], G[B]$ are connected. Then, for any $d \in C$,*

$$m(G, \omega, d) \leq m(A, \omega, d) + m(B, \omega, d).$$

A key step used to prove Theorem 3.2 in [16] is to prove a special case of Corollary 3.3, where the underlying graph G is a tree and A and B are disjoint. We will need the following result, proved using an extension of part of this proof from [16].

Lemma 3.4. *Let T be a tree, with colouring ω from colour-set C, let A and B be disjoint subsets of $V(T)$ such that $V(T) = A \cup B$ and $T[A], T[B]$ are connected, and let x be the unique vertex of B with a neighbour in A. Suppose that*

- *the sequence S_A floods $T[A]$ with colour d_A,*
- *the sequence S_B floods $T[B]$ with colour d_B,*
- *at least one move of S_B changes the colour of x, and*
- *playing S_A in T changes the colour of x.*

Then

$$m(T, \omega, d_B) \leq |S_A| + |S_B|.$$

Proof. We proceed by induction on $|B|$. Note that we may assume without loss of generality that ω gives a proper colouring of B; otherwise we may contract monochromatic components. Suppose $|B| = 1$. Then S_A must change the colour of the only vertex in B (linking it to some $a \in A$), and so playing S_A in T makes the whole tree monochromatic with colour d_A. Thus $m(T, \omega, d_A) \leq |S_A|$, and

$$m(T, \omega, d_B) \leq m(T, \omega, d_A) + 1 \leq |S_A| + 1 \leq |S_A| + |S_B|,$$

as required, since by assumption $|S_B| \geq 1$.

Now suppose $|B| > 1$, so B is not monochromatic initially, and assume that the result holds for smaller B. Set ${S_B}'$ to be the initial segment of S_B, up to and including the move that first makes B monochromatic (in any colour d'), so any final moves that simply change the colour of B are omitted. We may, of course, have ${S_B}' = S_B$ (and so $d' = d_B$), if B is not monochromatic before the final move of S_B.

Suppose that S'_B does not change the colour of x (which is only possible in the case $S'_B \neq S_B$). Then playing S'_B in T to make B monochromatic cannot change the colour of any vertex in A, so if we play S'_B in T and then play S_A, this will still flood A with colour d_A. Moreover, as playing S'_B has not changed the colour of x, playing S_A will still change the colour of x, thus linking all of B to A and so flooding T with colour d_A. Hence, in this case, we have

$$m(T, \omega, d_A) \leq |S'_B| + |S_A|,$$

and so, as we must in this case have $|S'_B| < |S_B|$,

$$m(T, \omega, d_B) \leq 1 + m(T, \omega, d_A) \leq 1 + |S'_B| + |S_A| \leq |S_A| + |S_B|,$$

as required.

Suppose now that S'_B does change the colour of x. Before the final move of ${S_B}'$ there are $r \geq 2$ monochromatic components in B (all but one of which have colour d'), with vertex-sets $B_1, \ldots, B_r$. For $1 \leq i \leq r$, set S_i to be the subsequence of ${S_B}'$ consisting of moves played in B_i, and note that these subsequences partition ${S_B}'$. Observe also that playing S_i in $T[B_i]$ gives B_i colour d', so $m(B_i, \omega, d') \leq |S_i|$.

Let B_1 be the unique component adjacent to A, and set $T_1 = T[A \cup B_1]$. Note that S_A floods $T_1[A]$ with colour d_A, and S_1 floods $T_1[B_1]$ with colour d'. Moreover, as playing S_A in T changes the colour of x, playing S_A in T_1 must also change the colour of x. Also, at least one move from S_B changes the colour of x, the unique vertex of B_1 with a neighbour in A, and this move must belong to S_1. Thus we can apply the inductive hypothesis to see that

$$m(T_1, \omega, d') \leq |S_A| + |S_1|.$$

Now suppose without loss of generality that B_2 is adjacent to B_1. We can then apply Corollary 3.3 to $T_2 = T[V(T_1) \cup B_2]$ to see that

$$m(T_2, \omega, d') \leq m(T_1, \omega, d') + m(B_2, \omega, d') \leq |S_A| + |S_1| + |S_2|.$$

Continuing in this way, each time adding an adjacent component, we see that

$$m(T, \omega, d') \leq |S_A| + \sum_{i=1}^{r} |S_i| = |S_A| + |{S_B}'|.$$

Now, if ${S_B}' = S_B$, this immediately gives the desired result, as $d' = d_B$. Otherwise, note that $|S_B| \geq |{S_B}'| + 1$ and so

$$m(T, \omega, d_B) \leq m(T, \omega, d') + 1 \leq |S_A| + |{S_B}'| + 1 \leq |S_A| + |S_B|,$$

as required. □

In the next result, we exploit this lemma to give a strengthening of Corollary 3.3 under additional assumptions. This can be applied to show that, in certain situations, we may assume that *no* optimal sequence to flood a subtree can change the colour of any vertex outside the subtree, when played in a larger tree.

Proposition 3.5. *Let T be a tree, with colouring ω from colour-set C, and let X and Y be disjoint sutbrees of T such that $T[V(X) \cup V(Y)]$ is connected, and such that*

- *there is a sequence S_X of α moves that floods X with some colour $d' \in C$,*
- *there is a sequence S_Y of β moves that floods Y with colour d, and that changes the colour of the unique vertex $v \in V(B) \cap \Gamma(A)$, and*
- *playing S_X in T changes the colour of at least one vertex in Y.*

Then, setting $T' = T \setminus (V(X) \cup V(Y))$ and $\omega' = \omega$, we have

$$m(T, \omega, d) \leq m(T', \omega', d) + \alpha + \beta.$$

Proof. Note that S_X must change the colour of v, so we can apply Lemma 3.4 to see that

$$m(T[V(X) \cup V(Y)], \omega, d) \leq |S_X| + |S_Y| = \alpha + \beta.$$

Corollary 3.3 then gives

$$\begin{aligned} m(T, \omega, d) &\leq m(T[V(X) \cup V(Y)], \omega, d) + m(T', \omega', d) \\ &\leq m(T', \omega', d) + \alpha + \beta, \end{aligned}$$

as required. □

Before proving the main result of this section, we need one further result, relating the number of moves required to flood the same graph with different initial colourings.

Lemma 3.6. *Let G be a connected graph, and let ω and ω' be two colourings of the vertices of G (from colour-set C). Let $\mathcal{A}$ be the set of all monochromatic components of G with respect to ω', and for each $A \in \mathcal{A}$ let c_A be the colour of A under ω'. Then, for any $d \in C$,*

$$m(G, \omega, d) \leq m(G, \omega', d) + \sum_{A \in \mathcal{A}} m(A, \omega, c_A).$$

Proof. We proceed by induction on $m(G, \omega', d)$. Note that if $m(G, \omega', d) = 0$ then the result is trivially true: in this case $\mathcal{A}$ contains a single monochromatic component G, with colour d, so we have

$$m(G, \omega', d) + \sum_{A \in \mathcal{A}} m(A, \omega, c_A) = m(G, \omega, d).$$

Suppose now that $m(G, \omega', d) > 0$, and let S be an optimal sequence of moves to flood G with colour d, when the initial colouring is ω'. We proceed by case analysis on the final move, α, of S. First suppose that G is already monochromatic before α, so this final move just changes the colour of the entire graph to d from some colour $d' \in C$. In this case, $m(G, \omega', d) = m(G, \omega', d') + 1$, and so we may apply the inductive hypothesis to see that

$$\begin{aligned} m(G, \omega, d) &\leq 1 + m(G, \omega, d') \\ &\leq 1 + m(G, \omega', d') + \sum_{A \in \mathcal{A}} m(A, \omega, c_A) \\ &= m(G, \omega', d) + \sum_{A \in \mathcal{A}} m(A, \omega, c_A), \end{aligned}$$

as required.

Now suppose that G is not monochromatic before α, and so this move links monochromatic components $X_1, \ldots, X_r$. We may assume that α changes the colour of X_1 from d' to d, and that all the components $X_2, \ldots, X_r$ have colour d before α. Let S_i denote the subsequence of S consisting of moves played in X_i, and observe that playing S_i in the isolated subgraph X_i must flood this graph with colour d, so $m(X_i, \omega', d) \leq |S_i|$. Note that, as no move can split a monochromatic component, the sets $\mathcal{A}_i = \{A \in \mathcal{A} : A \subseteq X_i\}$ (for $1 \leq i \leq r$) partition $\mathcal{A}$.

Observe that, for $2 \leq i \leq r$, $m(X_i, \omega, d) < |S| = m(G, \omega', d)$, and so we may apply the inductive hypothesis to see that

$$\begin{aligned} m(X_i, \omega, d) &\leq m(X_i, \omega', d) + \sum_{A \in \mathcal{A}_i} m(A, \omega, c_A) \\ &\leq |S_i| + \sum_{A \in \mathcal{A}_i} m(A, \omega, c_A). \end{aligned}$$

Similarly, the inductive hypothesis gives

$$m(X_1, \omega, d') \leq m(X_1, \omega', d') + \sum_{A \in \mathcal{A}_1} m(A, \omega, c_A),$$

and so, as $m(X_1, \omega', d') \leq |S_1| - 1$, we have

$$\begin{aligned} m(X_1, \omega, d) &\leq 1 + m(X_1, \omega, d) \\ &\leq |S_1| + \sum_{A \in \mathcal{A}_1} m(A, \omega, c_A). \end{aligned}$$

Now we can apply Corollary 3.3 to see that

$$m(G, \omega, d) \leq \sum_{i=1}^{r} m(X_i, \omega, d),$$

and so

$$\begin{aligned} m(G, \omega, d) &\leq \sum_{i=1}^{r} (|S_i| + \sum_{A \in \mathcal{A}_i} m(A, \omega, c_A)) \\ &= |S| + \sum_{A \in \mathcal{A}} m(A, \omega, c_A) \\ &= m(G, \omega', d) + \sum_{A \in \mathcal{A}} m(A, \omega, c_A), \end{aligned}$$

completing the proof. □

Using the previous results, we are now ready to prove the key result of this section.

Lemma 3.7. *Let G with colouring ω (from colour-set C) be the graph corresponding to a $2 \times n$ flood-it board B, let H be a connected induced subgraph of G, and let u and w be vertices lying in the leftmost and rightmost columns of H respectively. Then, for any $d \in C$, there exists a d-minimal spanning tree T for H such that* $\text{trunk}(T) \subseteq P(T, u, w)$.

Proof. We proceed by induction on $m(H, \omega, d)$. Note that the result is trivially true if $m(H, \omega, d) = 0$ as the graph is initially monochromatic with colour d and so any spanning tree will do. Suppose then that $m(H, \omega, d) > 0$. Let S be an optimal sequence to flood H with colour d, and suppose that the last move of S is α.

If H is monochromatic in some colour $d' \in C$ before α is played, and so this final move just changes the colour of the whole graph to d, we see that $m(H, \omega, d') \leq m(H, \omega, d) - 1$. Thus we may apply the inductive hypothesis to obtain a d'-minimal spanning tree T for H such that $\text{trunk}(T) \subseteq P(T, u, w)$. But then

$$m(T, \omega, d) \leq 1 + m(T, \omega, d') = 1 + m(H, \omega, d') \leq m(H, \omega, d),$$

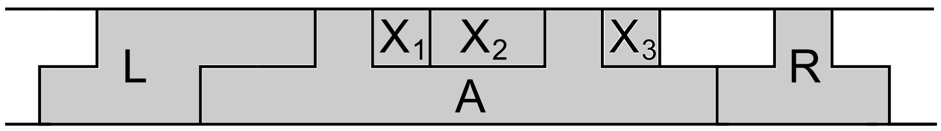

Figure 1: Monochromatic components of H before the final move is played.

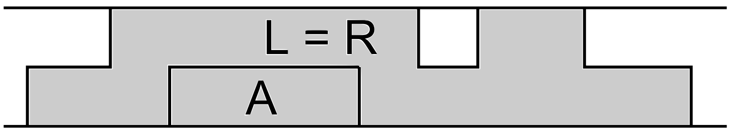

Figure 2: It is possible that $L = R$.

and so T is also a d-minimal spanning tree for H.

Thus we may assume that H is not monochromatic immediately before α is played. This means that α must change the colour of a monochromatic component A from some $d' \in C$ to d, where $H \setminus A$ is nonempty and has colour d before α is played. Since H is a connected induced subgraph of a $2 \times n$ board, $H \setminus A$ has at most one component L which contains vertices lying in columns to the left of all columns containing a vertex of A, and correspondingly at most one component R containing vertices lying in columns entirely to the right of A. There may additionally be some components $X_1, \ldots, X_r$ of $H \setminus A$ which contain only vertices which lie in the same column as some vertex of A. A possible structure for H is illustrated in Figure 1. In the remainder of the proof, we will exploit the structure of $H \setminus A$ to define a d-minimal spanning tree T for H whose non-leaf vertices lie on $P(T, u, w)$.

Observe that we may have $L = R$, as illustrated in Figure 2; we will deal with this case later, so for the moment we assume that $L \neq R$.

Set v (respectively v') to be any vertex lying in the leftmost (respectively rightmost) column of A that has at least one neighbour in L (respectively R); if L (respectively R) is empty, we set $v = u$ (respectively $v' = w$). If two vertices of L lie in the rightmost column of L, one of these must be adjacent to v, in which case we set this vertex to be u'; otherwise u' is defined to be the unique vertex of L that lies in the rightmost column. We define w' symmetrically, so that w' lies in the leftmost column of R, and so that if there is a choice of vertices of R in this column then w' is the vertex adjacent to v'. Note that $m(L, \omega, d) < m(H, \omega, d)$ and so, by the inductive hypothesis, there exists a d-minimal spanning tree T_L for L such that $\text{trunk}(T_L) \subseteq P(T_L, u, u')$. Similarly, there exists a d-minimal spanning tree T_R for R such that $\text{trunk}(T_R) \subseteq P(T_R, w', w)$, and a d'-minimal spanning tree T_A for A such that $\text{trunk}(T_A) \subseteq P(T_A, v, v')$. Let S_A be an optimal

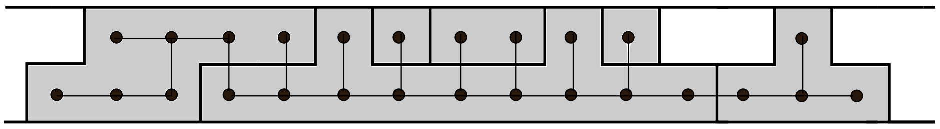

Figure 3: The spanning tree T.

sequence of moves to flood T_A with colour d', and S_L and S_R be optimal sequences to flood T_L and T_R respectively with colour d.

Observe that, as well as containing vertices that lie in columns to the left (respectively right) of A, L (respectively R) may additionally contain some vertices that lie in the same column as a vertex of A. We set T'_L to be the subtree of T_L induced only by those vertices in L that lie in the same column as or to the left of the leftmost vertex of A, and define T'_R symmetrically. We further define S'_L (respectively S'_R) to be the subsequence of S_L (respectively S_R) consisting of moves that change the colour of at least one vertex in T'_L (respectively T'_R), and note then that S'_L (respectively S'_R) floods T'_L (respectively T'_R) with colour d (implying that $m(T'_L, \omega, d) \leq |S'_L|$, and $m(T'_R, \omega, d) \leq |S'_R|$).

Now set T'_A to be the spanning tree for $H \setminus (T'_L \cup T'_R)$ obtained from T_A by adding an edge from every vertex z of this subgraph that does *not* lie in A to the vertex of A that lies in the same column as z (and observe that $\text{trunk}(T'_A) \subseteq P(T'_A, v, v')$). Finally, we obtain a spanning tree T for H by connecting T'_L, T'_R and T'_A. If $T'_L = T_L$, we use the edge $u'v$ to connect T'_L and T'_A; otherwise we use the edge of T_L with exactly one endpoint in T'_L. In either case we must have $\text{trunk}(T[V(T'_L) \cup V(T'_A)]) \subseteq P(T, u, v')$. Similarly, if $T'_R = T_R$ then we connect T'_R and T'_A with $v'w'$, and otherwise use the edge of T_R with exactly one endpoint in T'_R. The construction of T is illustrated in Figure 3. It is clear from the construction that T is a spanning tree for H, and that $\text{trunk}(T) \subseteq P(T, u, w)$; we will argue that in fact T is a d-minimal spanning tree for H.

For the rest of the argument, it will be useful to identify two important vertices of T. We set x to be the last vertex on the path $P(T, u, w)$ before A, and y the first vertex after A, when this path is traversed from left to right (as illustrated in Figure 3); if L (respectively R) is empty then x (respectively y) is not defined. Assuming L (respectively R) is nonempty, let a_x (respectively a_y) be the neighbour in A of x (respectively y). Note that $x \in T'_L$ and $y \in T'_R$.

Having defined the spanning tree T for H, we now consider how to flood

T with colour d. First, observe that

$$\begin{aligned} |S| &\geq 1 + m(A,\omega,d') + m(L,\omega,d) + m(R,\omega,d) \\ &\qquad + \sum_{i=1}^{r} m(X_i,\omega,d) \\ &\geq 1 + |S_A| + |S_L| + |S_R| + |\bigcup_{i=1}^{r} \operatorname{col}(X_i,\omega) \setminus \{d\}|. \end{aligned} \tag{1}$$

We will say that a colour $\tilde{d} \in \operatorname{col}(V(T'_A) \setminus V(A), \omega) \setminus \{d\}$ is *autonomous* either if $\tilde{d}$ appears in the initial colouring in one or more component X_i, or else if a move of S_L or S_R is is played in a monochromatic component of colour $\tilde{d}$ that does not intersect T'_L or T'_R. Let C_A denote the set of all autonomous colours. Note then that, for each $v \in V(T_L \setminus T'_L)$ that does not have colour d initially, at least one of the following must hold in order for v to be given colour d:

1. $\operatorname{col}(\{v\},\omega) \in C_A$, or
2. either initially, or after some move of S_L, x has colour $\operatorname{col}(\{v\},\omega)$.

Let W_L be the set of vertices $v \in V(T_L \setminus T'_L)$ such that the first statement holds, so $v \in W_L$ if and only if $\operatorname{col}(\{v\},\omega) \in C_A$. We then set $U_L = V(T_L \setminus T'_L) setminus W_L))$, and note that the second statement must hold for every $v \in U_L$. We can apply exactly the same reasoning to $V(T_R \setminus T'_R)$ (replacing x with y), and define U_R and W_R analogously.

Observe that $|S_L| \geq |S'_L| + |\operatorname{col}(W_L,\omega)|$, and $|S_R| \geq |S'_R| + |\operatorname{col}(W_R,\omega)|$. Thus, by (1), we see that

$$\begin{aligned} |S| &\geq 1 + |S_A| + |S'_L| + |S'_R| + |\operatorname{col}(W_L,\omega)| + |\operatorname{col}(W_R,\omega)| \\ &\qquad + |\bigcup_{i=1}^{r} \operatorname{col}(X_i,\omega) \setminus \{d\}| \\ &\geq 1 + |S_A| + |S'_L| + |S'_R| + |C_A|. \end{aligned}$$

In order to flood T with colour d, we will first play S_A, flooding A, and then repeatedly change the colour of A to cycle through all colours in C_A. Note that these first $|S_A|+|C_A|$ moves create a monochromatic component A' containing $T'_A \setminus (U_L \cup U_R)$. There are now three cases to consider, depending on whether none, both or one of U_L and U_R are non-empty.

First suppose that $U_L = U_R = \emptyset$. Note in this case that our first $|S_A|+|C_A|$ moves make T'_A monochromatic in some colour, so $m(T'_A,\omega,d) \leq 1 + |S_A| +$

$|C_A|$. Thus we can apply Corollary 3.3 to see that

$$\begin{aligned} m(T,\omega,d) &\leq m(T'_L,\omega,d) + m(T'_A,\omega,d) + m(T'_R,\omega,d) \\ &\leq 1 + |S_A| + |S'_L| + |S'_R| + |C_A| \\ &\leq |S| \\ &= m(H,\omega,d), \end{aligned}$$

as required.

Now suppose that exactly one of U_L and U_R is nonempty, and without loss of generality suppose that $U_L \neq \emptyset$. We claim that playing S_A in T does not change the colour of any vertex in T'_L. Indeed, if this sequence does change the colour of a vertex in T'_L, it must change the colour of x, and this colour change will be due to moves in S_A changing the colour of a_x. Thus, if we played S_A in the tree T_1, obtained by connecting T_L, T_A and T_R with the edges xa_x and ya_y, it would would still change the colour of x (which is the unique vertex of T_L adjacent to T_A). However, as $U_L \neq \emptyset$, we know that S_L must change the colour of x, and so by Proposition 3.5 (setting $X = T_A$, $Y = T_L$, $S_X = S_A$ and $S_Y = S_L$) we would have $m(T_1,\omega,d) \leq |S_R| + |S_L| + |S_A| < |S|$, implying (by Theorem 3.2) that $m(H,\omega,d) \leq m(T_1,\omega,d) < |S| = m(H,\omega,d)$, a contradiction.

We may further assume that then cycling A through all colours in C_A does not change the colour of any vertex in T'_L: if $\text{col}(\{x\},\omega) \in C_A$ we can choose this to be the last colour we play in A, and so our sequence will link A to x but will not change the colour of x (or therefore of any other vertex in T'_L).

Next, if playing S_A and cycling through the colours of C_A has not already linked A' to x, we play one further move to give A' the same colour as x. Since the sequence of moves we play up to this point does not change the colour of any vertex in T'_L, we can now play the sequence S'_L to give every vertex in T'_L colour d. As x is in the same monochromatic component as A' this will also give all vertices in A' colour d. Moreover, playing this sequence will at some point give x, and hence A', every colour in $\text{col}(U_R,\omega)$, and so will link every vertex in U_R to A' and ultimately give these vertices colour d. Thus, playing S_A, cycling through C_A, if necessary linking x to A', and then playing S'_L will flood all the vertices of $T \setminus T'_R$ with colour d (as $U_R = \emptyset$), so we see that

$$m(T \setminus T'_R,\omega,d) \leq |S_A| + |C_A| + 1 + |S'_L|.$$

But then, once again, we can apply Corollary 3.3 to see that

$$\begin{aligned} m(T,\omega,d) &\leq m(T\setminus T'_R,\omega,d) + m(T'_R,\omega,d) \\ &\leq 1 + |S_A| + |C_A| + |S'_L| + |S'_R| \\ &\leq |S| \\ &= m(H,\omega,d), \end{aligned}$$

as required.

For the final subcase, we suppose that $U_L, U_R \neq \emptyset$. We begin once again by playing S_A, cycling A through all colours in C_A, and then (if required) playing an additional move to change the colour of the monochromatic component containing A to be the same as x; as before we may assume that these initial moves do not change the colour of any vertex in S'_L.

Note that, as $U_L \neq \emptyset$, the colour of x must change at least once when we play S'_L in T'_L. Set β to be the last move in S'_L to change the colour of x, and note then that β must change the colour of some component Z, containing x, to d. Set $\bar{T}_L = T'_L \setminus Z$, and let S_Z be the subsequence of S'_L consisting of moves played in Z (so S_Z floods Z with colour d, and β is the final move of S_Z). As Z is monochromatic before β, playing $S_Z \setminus \beta$ in Z must flood this component with some colour $d_Z \in C$. Observe also that the sequence $S'_L \setminus S_Z$ must, when played in the forest $\bar{T}_L$, give every vertex of $\bar{T}_L$ colour d.

Suppose that, after playing S_A and linking A to x, we then play $S_Z \setminus \beta$. This will ensure that x and hence A at some point receives every colour in $\text{col}(U_L,\omega)$ (except possibly d), so every vertex in U_L is either linked to A or has colour d (in which case it will certainly end up with colour d, as its colour can only change if it is linked to another vertex which will ultimately be given colour d). Note that we now have a monochromatic component B that contains A, Z and all vertices of $T'_A \setminus U_R$ that do not initially have colour d.

We claim that the sequence of moves we play up to this point cannot change the colour of any vertex in T'_R. To prove the validity of this claim, first observe that S'_R, played in T_R, floods a subtree T''_R of T_R with colour d, where $T'_R \cup U_R \subseteq T''_R$. Now set T_2 to be the spanning tree for H obtained by connecting T''_R and $T \setminus V(T''_R)$ with the edge ya_y. It is clear that, if our sequence of moves so far changes the colour of any vertex in T'_R when played in T, playing the same sequence in T_2 would change the colour of $y \in T''_R$. However, as $U_R \neq \emptyset$, we also know that S_R changes the colour of y. Note also that all vertices of $T_2 \setminus (B \cup T''_R)$ that do not belong to $\bar{T}_L$ have colour d initially, so $m(\bar{T}_L,\omega,d)$ moves suffice to flood $T_2 \setminus (B \cup T''_R)$ with colour d. We can now apply Proposition 3.5, setting $X = B \setminus T''_R$, S_X to be the sequence of moves we have played up to this point, $Y = T''_R$ and $S_Y = S'_R$ to see that

$$\begin{aligned}
m(T_2, \omega, d) &\leq m(\bar{T}_L, \omega, d) + |S_A| + 1 + |S_Z| - 1 + |C_A| + |S'_R| \\
&\leq |S'_L| - |S_Z| + |S_A| + |S_Z| + |C_A| + |S'_R| \\
&= |S'_L| + |S'_R| + |S_A| + |C_A| \\
&< |S|.
\end{aligned}$$

Theorem 3.2 would then imply that

$$m(H, \omega, d) \leq m(T_2, \omega, d) < |S| = m(H, \omega, d),$$

a contradiction.

If the monochromatic component containing A does not already have the same colour as y, we now play one further move to link it to this vertex (and note that such a move will not change the colour of any vertex in T'_R). Hence, if we now play S'_R, this will flood T'_R with colour d; as A and y lie in the same monochromatic component before these moves are played, this sequence will also give every vertex in the same monochromatic component as A colour d. Moreover, linking A to y and playing S'_R will at some point give y, and hence A, every colour in $\text{col}(U_R, \omega)$, and so all vertices in U_R will be linked to A and thus end up with colour d. Hence this sequence of moves gives every vertex in $T \setminus \bar{T}_L$ colour d, and so we have

$$\begin{aligned}
m(T \setminus \bar{T}_L, \omega, d) &\leq |S_A| + |C_A| + 1 + |S_Z| - 1 + 1 + |S'_R| \\
&= |S_A| + |C_A| + |S_Z| + |S'_R| + 1.
\end{aligned}$$

Finally, we apply Corollary 3.3 to give

$$\begin{aligned}
m(T, \omega, d) &\leq m(T \setminus \bar{T}_L, \omega, d) + m(\bar{T}_L, \omega, d) \\
&\leq |S_A| + |C_A| + |S_Z| + |S'_R| + 1 + |S'_L| - |S_Z| \\
&= |S_A| + |C_A| + |S'_R| + |S'_L| + 1 \\
&\leq |S| \\
&= m(H, \omega, d),
\end{aligned}$$

as required. This completes the proof in the case that $L \neq R$.

It remains to consider the case in which $L = R$, as in Figure 2. We define T exactly as before (as shown in Figure 4), and again identify the important vertices x and y. The previous reasoning only fails in the case $L = R$ because it is not necessarily true, in this case, that S'_L floods T'_L with colour d and S'_R floods T'_R with colour d. However, by considering more carefully the sequence of moves that floods $H \setminus A$, we are able to deal with this problem.

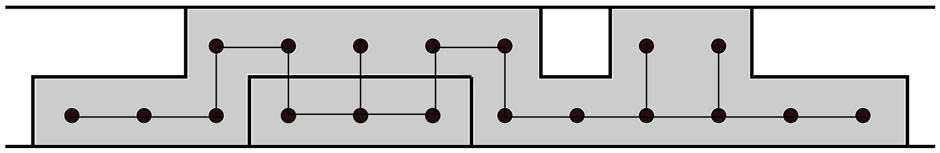

Figure 4: The construction of T in the case that $L = R$.

If x and y belong to the same monochromatic component T' of $T_L(= T_R)$, with colour d_{xy}, under the initial colouring ω, then we can flood $T[V(T') \cup V(A)]$ by playing S_A and then changing the colour of A to d_{xy}, so $m(T[V(T') \cup V(A)], \omega, d_{xy}) \leq |S_A| + 1$. Let ω' be the colouring of T which agrees with ω on every vertex in T_L, and gives every vertex in A colour d_{xy}. Then T with colouring ω' is equivalent (when monochromatic components are contracted) to T_L with colouring ω, implying that $m(T, \omega', d) = m(T_L, \omega, d) \leq |S_L|$. We can then apply Lemma 3.6 to give

$$\begin{aligned} m(T, \omega, d) &\leq m(T, \omega', d) + m(T[V(T') \cup V(A)], \omega, d_{xy}) \\ &\leq |S_L| + |S_A| + 1 \\ &= |S| \\ &= m(H, \omega, d). \end{aligned}$$

So we may assume that x and y do not belong to the same monochromatic component initially. Let S' be the initial segment of S_L up to and including the move that first links x and y; let T' be the monochromatic component of $T \setminus A$ that contains x and y at this point, and suppose that it has colour $\bar{d}$ and that k moves of S' are played in T'.

We now consider flooding the subtree $T[V(T') \cup V(A)]$ with colour $\bar{d}$. Note that the subsequence of S' consisting of moves played in $T'_L \cap T'$ must in this case flood $T'_L \cap T'$ with colour $\bar{d}$, and similarly the subsequence consisting of moves played in $T'_R \cap T'$ must flood $T'_R \cap T'$ with colour $\bar{d}$. Thus, applying exactly the same arguments as in the case that $L \neq R$, we see that

$$m(V(T') \cup V(A), \omega, \bar{d}) \leq |S_A| + k + 1.$$

Now set ω' to be the colouring of $V(T)$ that agrees with $S'(\omega, T_L)$ on T_L and gives all vertices of A colour $\bar{d}$. Note that T with colouring ω' is equivalent (when monochromatic components are contracted) to T_L with colouring $S'(\omega, T_L)$, and so $m(T, \omega', d) = m(T_L, S'(\omega, T_L), d) \leq |S_L| - |S'|$. If $\mathcal{A}$ is the set of monochromatic components of T with respect to ω', and each $A \in \mathcal{A}$ has colour c_A under this colouring, then observe that

$$\sum_{\substack{A \in \mathcal{A} \\ A \neq T[V(T') \cup V(A)]}} m(A, \omega, c_A) \leq |S'| - k,$$

and so

$$\sum_{A \in \mathcal{A}} m(A, \omega, c_A) \leq |S'| - k + |S_A| + k + 1 = |S'| + |S_A| + 1.$$

We can now apply Lemma 3.6 to see that

$$\begin{aligned} m(T, \omega, d) &\leq m(T, \omega', d) + \sum_{A \in \mathcal{A}} m(A, \omega, c_A) \\ &\leq |S_L| - |S'| + |S'| + |S_A| + 1 \\ &= |S| \\ &= m(H, \omega, d) \end{aligned}$$

in this case also, completing the proof. □

In our analysis of the algorithm in the next section, we will need one additional result: we show in the next lemma that any tree can be flooded by an optimal sequence in which no moves are played at leaves.

Lemma 3.8. *Let T be any tree, and ω a colouring of the vertices of T. Then there exists a sequence of moves S, of length $m(T, \omega)$, which makes T monochromatic and in which all moves are played in* $\text{trunk}(T)$.

Proof. Let S_0 be any optimal sequence to flood T, and set S_0' to be the subsequence of S_0 consisting of moves that change the colour of a vertex in $\text{trunk}(T)$. Note that we may assume without loss of generality that all moves of S_0' are played in $\text{trunk}(T)$. Note further that $S_0 \setminus S_0'$ contains only moves played at leaves, and let U be the set of leaves in which moves of $S_0 \setminus S_0'$ are played. Observe that playing S_0' in T will make $T \setminus U$ monochromatic, and so we can flood the entire tree by playing a sequence S which consists of S_0' followed by a further $|\operatorname{col}(U, \omega)|$ moves, cycling through the colours still present in leaves of T (playing all moves in $\text{trunk}(T)$). Thus

$$|S_1| \leq |S_0'| + |\operatorname{col}(U, \omega)| \leq |S_0'| + |U|.$$

However, it is clear that $|S_0| \geq |S_0'| + |U|$, as $S_0 \setminus S_0'$ contains at least one move played at each vertex in U. Hence we see that $|S| \leq |S_0|$, and so S is an optimal sequence to flood T in which all moves are played in $\text{trunk}(T)$, as required. □

3.2 The algorithm

In this section we describe our algorithm to solve c-FREE-FLOOD-IT on $2 \times n$ boards, and use results from the previous section to prove its correctness.

We begin with some further definitions. For any section $B[b_1, b_2]$, we define $\mathcal{T}[b_1, b_2]$ to be the set of all spanning trees for $B[b_1, b_2]$. Given any $2 \times n$ Flood-It board B, corresponding to a graph G with colouring ω from colour-set C, we define a set of vectors $Z(B)$, where

$$\begin{aligned} Z(B) = \{(b_1, b_2, r_1, r_2, d, I) :& \\ & B[b_1, b_2] \text{ is a section}, \\ & r_1, r_2 \in V(B[b_1, b_2]), \\ & \exists T \in \mathcal{T}[b_1, b_2] \text{ such that } \operatorname{trunk}(T) \subseteq P(T, r_1, r_2), \\ & r_1 \text{ incident with } b_1, r_2 \text{ incident with } b_2, \\ & d \in C, \\ & I \subseteq C\}. \end{aligned}$$

Note that there always exists a tree $T \in \mathcal{T}[b_1, b_2]$ such that $\operatorname{trunk}(T) \subseteq P(T, r_1, r_2)$ unless one of the following holds:

1. there is more than one vertex of $B[b_1, b_2]$ lying strictly to the left of r_1 or strictly to the right of r_2, or

2. there is exactly one vertex of $B[b_1, b_2]$ lying strictly to the left of r_1 (respectively to the right of r_2), which is not adjacent to r_1 (respectively r_2) and whose neighbour in the same column as r_1 (respectively r_2) has no neighbour in $B[b_1, b_2]$ other than r_1 (respectively r_2).

Thus we can check whether this condition is satisfied in constant time.

We now introduce a function f which is closely related to the minimum number of moves required to flood a $2 \times n$ board. For any $\mathbf{z} = (b_1, b_2, r_1, r_2, d, I) \in Z(B)$ we define $f(\mathbf{z})$ to be the minimum, taken over all $T \in \mathcal{T}[b_1, b_2]$ such that $\operatorname{trunk}(T) \subseteq P(T, r_1, r_2)$, of the number of moves that must be played in $P(T, r_1, r_2)$ to flood $P(T, r_1, r_2)$ with colour d, and link to $P(T, r_1, r_2)$ all leaves of T that do not have colours from I.

It follows immediately from Lemmas 3.7 and 3.8 that

$$m(G, \omega) = \min_{\substack{d \in C \\ r_1 \text{ incident with } b_L \\ r_2 \text{ incident with } b_R}} f(b_L, b_R, r_1, r_2, d, \emptyset).$$

Our algorithm in fact computes recursively a function f^*, with the same parameters as f. We will argue that, for every $\mathbf{z} \in Z(B)$, $f^*(\mathbf{z}) = f(\mathbf{z})$

and hence that it suffices to compute all values of f^* in order to calculate $m(G, \omega)$.

The first step of the algorithm is to initialise certain values of f^* to zero. We set $f^*(b_1, b_2, r_1, r_2, d, I) = 0$ if and only if, under the initial colouring, there exists a r_1-r_2 path of colour d in $B[b_1, b_2]$, and all vertices in $B[b_1, b_2]$ that do not lie on this path are adjacent to the path and have colours from $I \cup \{d\}$. All other values of $f^*(\mathbf{z})$ are initially set to infinity. Note that, under this definition, $f^*(\mathbf{z}) = 0 \iff f(\mathbf{z}) = 0$, and that for each $\mathbf{z} \in Z(B)$ we can easily determine in time $O(n)$ whether f^* should be initialised to zero or infinity.

In order to define further values of f^*, we introduce two more functions. First, for any $\mathbf{z} = (b_1, b_2, r_1, r_2, d, I) \in Z(B)$, we set

$$f_1(b_1, b_2, r_1, r_2, d, I) = 1 + \min_{d' \in C}\{f^*(b_1, b_2, r_1, r_2, d', I \cup \{d\})\}.$$

We also define, for any $\mathbf{z} \in Z(B)$,

$$f_2(b_1, b_2, r_1, r_2, d, I) = \min_{\substack{(b_1, b, r_1, x_1, d, I) \in Z(B) \\ (b, b_2, x_2, r_2, d, I) \in Z(B) \\ b_1 < b < b_2 \\ x_1 x_2 \in E(G)}} \{f^*(b_1, b, r_1, x_1, d, I) + f^*(b, b_2, x_2, r_2, d, I)\}.$$

Finally, we set

$$f^*(\mathbf{z}) = \min\{f_1(\mathbf{z}), f_2(\mathbf{z})\}.$$

For the reasoning below, it will be useful to introduce another function θ, taking the same parameters as f and f^*. For any $\mathbf{z} = (b_1, b_2, r_1, r_2, d, I) \in Z(B)$, we define

$$\theta(\mathbf{z}) = f^*(\mathbf{z}) + |\{b : b \text{ a border of } B, b_1 < b < b_2\}|.$$

In the following two lemmas, we show that $f^*(\mathbf{z}) = f(\mathbf{z})$ for all $\mathbf{z} \in Z(B)$, as claimed. We begin by demonstrating that $f^*(\mathbf{z})$ gives an upper bound for $f(\mathbf{z})$.

Lemma 3.9. *Let G with colouring ω (from colour-set C) be the coloured graph corresponding to a $2 \times n$ Flood-It board B. Then*

$$f(\mathbf{z}) \leq f^*(\mathbf{z})$$

for all $\mathbf{z} = (b_1, b_2, r_1, r_2, d, I) \in Z(B)$.

Proof. We proceed by induction on $\theta(\mathbf{z})$. Recall that we have equality between $f(\mathbf{z})$ and $f^*(\mathbf{z})$ whenever $f^*(\mathbf{z}) = 0$, so certainly the base case for $\theta(\mathbf{z}) = 0$ must hold. Assume therefore that $f^*(\mathbf{z}) > 0$, and that the result holds for all $\mathbf{z}'$ with $\theta(\mathbf{z}') < \theta(\mathbf{z})$.

Since $f^*(\mathbf{z}) > 0$, we must have $f^*(\mathbf{z}) \in \{f_1(\mathbf{z}), f_2(\mathbf{z})\}$. Suppose first that $f^*(\mathbf{z}) = f_1(\mathbf{z})$. Then, for some $d' \in C$,

$$\begin{aligned} f^*(b_1, b_2, r_1, r_2, d, I) &= 1 + f^*(b_1, b_2, r_1, r_2, d', I \cup \{d\}) \\ &\qquad\qquad\qquad \text{by definition of } f_1 \\ &\geq 1 + f(b_1, b_2, r_1, r_2, d', I \cup \{d\}) \\ &\qquad\qquad\qquad \text{by inductive hypothesis.} \end{aligned}$$

But then we know, by definition of f, that there exists $T \in \mathcal{T}[b_1, b_2]$ and a sequence S of $f(b_1, b_2, r_1, r_2, d', I \cup \{d\})$ moves, all played in $P(T, r_1, r_2)$, which, when played in T, floods $P(T, r_1, r_2) \supseteq \text{trunk}(T)$ with colour d' and links all leaves to $P(T, r_1, r_2)$ except possibly those with colours from $I \cup \{d\}$. By appending one further move to S, which changes the colour of $P(T, r_1, r_2)$ to d, we obtain a sequence S' of length $f(b_1, b_2, r_1, r_2, d', I \cup \{d\}) + 1$ (with all moves played in $P(T, r_1, r_2)$) which, when played in T, floods $P(T, r_1, r_2)$ with colour d and is such that all leaves of T not linked to $P(T, r_1, r_2)$ by S have colours from I. Hence $f(b_1, b_2, r_1, r_2, d, I) \leq |S'| = 1 + f(b_1, b_2, r_1, r_2, d', I \cup \{d\})$, and so $f(b_1, b_2, r_1, r_2, d, I) \leq f^*(b_1, b_2, r_1, r_2, d, I)$, as required.

Now suppose that $f^*(\mathbf{z}) = f_2(\mathbf{z})$. Then, by definition of f_2, there exists a border b with $b_1 < b < b_2$ and an edge $x_1x_2 \in E(G)$ such that $(b_1, b, r_1, x_1, d, I), (b, b_2, x_2, r_2, d, I) \in Z(B)$ and

$$f^*(b_1, b_2, r_1, r_2, d, I) = f^*(b_1, b, r_1, x_1, d, I) + f^*(b, b_2, x_2, r_2, d, I).$$

Note that $|\{b' : b' \text{ a border of } B, b_1 < b' < b\}|$ and $|\{b' : b \text{ a border of } B, b < b' < b_2\}|$ are both strictly smaller than $|\{b : b \text{ a border of } B, b_1 < b < b_2\}|$, so by the inductive hypothesis we have

$$f^*(b_1, b_2, r_1, r_2, d, I) \geq f(b_1, b, r_1, x_1, d, I) + f(b, b_2, x_2, r_2, d, I).$$

By definition of f, there exist trees $T_1 \in \mathcal{T}[b_1, b]$ and $T_2 \in \mathcal{T}[b, b_2]$, and sequences S_1 and S_2 of length $f(b_1, b, r_1, x_1, d, I)$ and $f(b, b_2, x_2, r_2, d, I)$ respectively, such that (for $i \in \{1, 2\}$) all moves of S_i are played in $P(T_i, r_i, x_i)$ and S_i floods $P(T_i, r_i, x_i)$ with colour d, additionally linking all leaves of T_i to $P(T_i, r_i, x_i)$ except possibly those with colours from I. Now set $T = T_1 \cup T_2 \cup \{x_1x_2\}$. It is clear that $T \in \mathcal{T}[b_1, b_2]$, and moreover that $\text{trunk}(T) \subseteq P(T, r_1, r_2)$. Suppose T_1' and T_2' are the subtrees of T_1 and T_2 respectively that are given colour d by S_1 and S_2, and set $T' = T_1' \cup T_2' \cup \{x_1x_2\}$. Note that

$P(T, r_1, r_2) \subseteq T'$ and that $\mathrm{col}(T \setminus T', \omega) = \mathrm{col}(T_1 \setminus T_1', \omega) \cup \mathrm{col}(T_2 \setminus T_2', \omega) \subseteq I$, so $f(b_1, b_2, r_1, r_2, d, I) \leq m(T', \omega, d)$. We can then apply Corollary 3.3 to see that

$$\begin{aligned} f(b_1, b_2, r_1, r_2, d, I) &\leq m(T', \omega, d) \\ &\leq m(T_1', \omega, d) + m(T_2', \omega, d) \\ &\leq |S_1| + |S_2| \\ &= f(b_1, b, r_1, x_1, d, I) + f(b, b_2, x_2, r_2, d, I) \\ &\leq f^*(b_1, b_2, r_1, r_2, d, I), \end{aligned}$$

completing the proof. □

Next we show that the reverse inequality also holds.

Lemma 3.10. *Let G with colouring ω (from colour-set C) be the coloured graph corresponding to a $2 \times n$ Flood-It board B. Then*

$$f(\mathbf{z}) \geq f^*(\mathbf{z})$$

for all $\mathbf{z} = (b_1, b_2, r_1, r_2, d, I) \in Z(B)$.

Proof. We proceed by induction on $f(\mathbf{z})$, noting again that we have equality in the base case for $f(\mathbf{z}) = 0$. Suppose that $f(\mathbf{z}) > 0$, and that the result holds for $\mathbf{z}'$ whenever $f(\mathbf{z}') < f(\mathbf{z})$. By definition, there exists a tree $T \in \mathcal{T}[b_1, b_2]$ and a sequence S of length $f(b_1, b_2, r_1, r_2, d, I)$ such that $\mathrm{trunk}(T) \subseteq P(T, r_1, r_2)$, all moves of S are played in $P(T, r_1, r_2)$, and S floods $P(T, r_1, r_2)$ with colour d, leaving only leaves with colours from I not linked to $P(T, r_1, r_2)$. We proceed by case analysis on α, the final move of S.

Suppose first that $P(T, r_1, r_2)$ is already monochromatic before α, and that this final move just changes its colour to d from some $d' \in C$ (possibly flooding some additional leaves of colour d in the process). In this case it is clear that $f(b_1, b_2, r_1, r_2, d', I \cup \{d\}) \leq |S| - 1$ and so we can apply the inductive hypothesis to see that $f^*(b_1, b_2, r_1, r_2, d', I \cup \{d\}) \leq f(b_1, b_2, r_1, r_2, d', I \cup \{d\})$. But then, by definition of f_1, we know that

$$\begin{aligned} f^*(b_1, b_2, r_1, r_2, d, I) &\leq 1 + f^*(b_1, b_2, r_1, r_2, d', I \cup \{d\}) \\ &\leq 1 + f(b_1, b_2, r_1, r_2, d', I \cup \{d\}) \\ &\leq 1 + |S| - 1 \\ &= f(b_1, b_2, r_1, r_2, d, I), \end{aligned}$$

as required.

So we may assume that $P(T, r_1, r_2)$ is not monochromatic before α: it may have either two or three monochromatic components. Suppose first that $P(T, r_1, r_2)$ has exactly three monochromatic components before α is played, A_1, A_2 and A_3; we may assume that A_1 and A_3 have colour d before α, and that this final move gives A_2 colour d to flood the entire path. For $i \in \{1, 2, 3\}$, set S_i to be the subsequence of $S \setminus \alpha$ consisting of moves played in A_i, and set $\bar{A}_i$ to be A_i together with all leaves of T that lie in the same column as a vertex of A_i or whose only neighbour on $P(T, r_1, r_2)$ is in A_i. Note that that $\bar{A}_1$, $\bar{A}_2$ and $\bar{A}_3$ partition the vertex set of T, and that S_1, S_2 and S_3 partition $S \setminus \alpha$. We may assume without loss of generality that $r_1 \in A_1$ and $r_2 \in A_3$. Observe that there must exist borders b and b', with $b_1 < b < b' < b_2$, such that $\bar{A}_1 = B[b_1, b]$, $\bar{A}_2 = B[b, b']$ and $\bar{A}_3 = B[b', b_2]$. Set $x_1 x_2$ to be the edge of T such that $x_1 \in A_1$, $x_2 \in A_2$ and $y_1 y_2$ the edge of T such that $y_1 \in A_2$ and $y_2 \in A_3$.

Note that $T[A_1] \in \mathcal{T}[b_1, b]$, $T[A_2] \in \mathcal{T}[b, b']$, and $T[A_3] \in \mathcal{T}[b', b_2]$, and moreover that we have $\text{trunk}(T[A_1]) \subseteq P(T[A_1], r_1, x_1)$, $\text{trunk}(T[A_2]) \subseteq P(T[A_2], x_2, y_1)$ and $\text{trunk}(T[A_3] \subseteq P(T[A_3], y_2, r_2)$. Observe also that S_1 is a sequence of moves played in $P(T[A_1], r_1, x_1)$ that floods $P(T[A_1], r_1, x_1)$ with colour d and links all leaves, except possibly those with colours from I, to $P(T[A_1], r_1, x_1)$, so we must have $f(b_1, b, r_1, x_1, d, I) \leq |S_1|$. Similarly, we see that $f(b', b_2, y_2, r_2, d, I) \leq |S_3|$ and $f(b, b', x_2, y_1, d', I \cup \{d\}) \leq |S_2|$. Since $|S_1|, |S_2|, |S_3| < |S|$, we can apply the inductive hypothesis to see that

$$f^*(b_1, b, r_1, x_1, d, I) \leq f(b_1, b, r_1, x_1, d, I) \leq |S_1|$$

and

$$f^*(b', b_2, y_2, r_2, d, I) \leq f(b', b_2, y_2, r_2, d, I) \leq |S_3|.$$

The inductive hypothesis also gives

$$f^*(b, b', x_2, y_1, d', I \cup \{d\}) \leq f(b, b', x_2, y_1, d', I \cup \{d\}) \leq |S_2|,$$

and we can then apply the definition of f_1 to see that

$$f^*(b, b', x_2, y_1, d, I) \leq 1 + f^*(b, b', x_2, y_1, d', I \cup \{d\}) \leq 1 + |S_2|.$$

Now we can apply the definition of f^* to see that

$$\begin{aligned} f^*(b_1, b_2, r_1, r_2, d, I) &\leq f_2(b_1, b_2, r_1, r_2, d, I) \\ &\leq f^*(b_1, b, r_1, x_1, d, I) + f^*(b, b_2, x_2, r_2, d, I) \\ &\leq f^*(b_1, b, r_1, x_1, d, I) + f_2(b, b_2, x_2, r_2, d, I) \\ &\leq f^*(b_1, b, r_1, x_1, d, I) + f^*(b, b', x_2, y_1, d, I) \\ &\qquad\qquad\qquad\qquad + f^*(b', b_2, y_2, r_2, d, I) \\ &\leq 1 + |S_1| + |S_2| + |S_3| \\ &= |S| \\ &= f(b_1, b_2, r_1, r_2, d, I), \end{aligned}$$

as required.

For the remaining case, in which $P(T, r_1, r_2)$ has exactly two monochromatic components before α, we can use the same reasoning as in the previous case for three components to show that we must once again have $f^*(b_1, b_2, r_1, r_2, d, I) \leq f(b_1, b_2, r_1, r_2, d, I)$, completing the proof. □

The final step to is to show that all values of f^* can be computed in time $O(n^{20}2^{3c})$.

Proposition 3.11. *For any $2 \times n$ Flood-It board B, the function $f^*(\mathbf{z})$ can be computed, for all $\mathbf{z} \in Z(B)$, in time $O(n^{11}2^c)$.*

Proof. We compute values of f^* recursively using a dynamic programming technique. Our table has one entry for each pair of borders, for each possible vertex incident with each of the borders, for each colour in the colour-set and for each possible subset of colours, so the total number of entries is at most

$$O(n^2 \cdot n^2 \cdot n \cdot n \cdot c \cdot 2^c) = O(n^6 2^c).$$

The table is initialised by setting all values to either zero or infinity, and for each entry we can determine which of these values it should take in time at most $O(n)$, so we can initialise the entire table in time $O(n^7 2^c)$.

The next step is to apply the recursive definition of f^* repeatedly to all entries in the table that are not already set to zero. Each time we apply this definition to a single entry, we take the minimum of at most $O(cn^3)$ values (one for each choice of colour, plus one for each combination of a border and a pair of adjacent vertices on either side), each a combination of at most two other entries in the table, so each entry can be calculated in time $O(cn^3)$. We can therefore perform one iteration in which we apply the definition to each non-zero entry in the table in time $O(n^9 2^c)$.

Note that once we have initialised the table, we have the correct value of $f^*(\mathbf{z})$ for any $\mathbf{z}$ such that $\theta(\mathbf{z}) = 0$. Moreover, the value of $f^*(\mathbf{z})$ depends only on values of $f^*(\mathbf{z}')$ where $\theta(\mathbf{z}') < \theta(\mathbf{z})$, so after k iterations we will have correctly computed the value of $f^*(\mathbf{z})$ for all $\mathbf{z}$ with $\theta(\mathbf{z}) \leq k$. Note that for every $\mathbf{z} \in Z(B)$, $\theta(\mathbf{z}) \leq 2n + n^2$, as there are at most n^2 borders in total, and no more than $2n$ moves can be required to flood a graph with at most this many vertices, so $n^2 + 2n$ iterations are sufficient to guarantee we have computed all values of f^* correctly.

Thus, we can compute all values of $f^*(\mathbf{z})$ for $\mathbf{z} \in Z(B)$ in time $O(n^{11}2^c)$, as required. □

We now combine the previous three results to give the proof of our main theorem.

Proof of Theorem 3.1. Recall that, from the definition of f and Lemmas 3.7 and 3.8,

$$m(G, \omega) = \min_{\substack{d \in C \\ r_1 \text{ incident with } b_L \\ r_2 \text{ incident with } b_R}} f(b_L, b_R, r_1, r_2, d, \emptyset).$$

Thus, in order to compute $m(G, \omega)$ in time $O(n^{11}2^c)$, it suffices to compute all relevant values of f in time $O(n^{11}2^c)$.

However, we know from Lemmas 3.9 and 3.10 that $f(\mathbf{z}) = f^*(\mathbf{z})$ for all $\mathbf{z} \in Z(B)$, and from Proposition 3.11 we know that we can compute $f^*(\mathbf{z})$ for all $\mathbf{z} \in Z(B)$ in time $O(n^{11}2^c)$. This completes the proof of the theorem. □

4 FREE FLOOD IT on $2 \times n$ boards

In this section we prove the following theorem.

Theorem 4.1. Free-Flood-It *remains NP-hard when restricted to* $2 \times n$ *boards.*

This is somewhat surprising, as we have seen in the previous section that c-Free-Flood-It can be solved in polynomial time on $2 \times n$ boards, while [7] gives a linear time algorithm to solve Fixed Flood It in this situation. We demonstrate here that the problem is almost certainly not in **P** if we remove both these restrictions (that moves are always played at the same vertex, or the number or colours is bounded). This is the first class of graphs for which such a result has been shown.

The proof is by means of a reduction from Vertex Cover, shown to be NP-hard by Karp in [11]. Given a graph $G = (V, E)$, we construct a $2 \times n$ Flood-It board B_G as follows.

Suppose $E = \{e_1, \ldots, e_m\}$. For each edge $e = uv \in E$ we construct the gadget G'_e, as illustrated in Figure 5. We will refer to the single-square components incident with the bottom edge in G'_e as *islands*. G'_e is then embedded in the larger gadget G_e, as shown in Figure 6. Distinct colours $x^e_1, \ldots, x^e_r$ are used for each e, where $r = 2m + |V|$. We then obtain the board B_G by placing these gadgets G_e in a row, as illustrated in Figure 7. Observe that we can take $n = m(2r + 6) = 2m(2m + |V| + 3)$. Let us also set $N = mr + 2m - 1$.

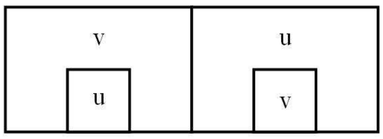

Figure 5: The gadget G'_e

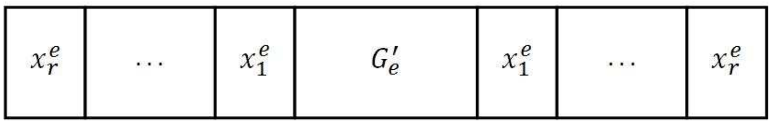

Figure 6: The gadget G_e

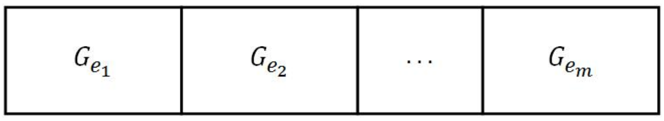

Figure 7: The board B

In the two following lemmas, we show that we can flood this board B in $N + k$ steps if and only if G has a vertex cover of size at most k.

Lemma 4.2. *If G has a vertex cover of size at most k, then we can flood the board B_G in $N + k$ steps.*

Proof. First observe that, if $e = uv$, then with $(r + 1)$ moves we can flood the gadget G_e, except for a single island of colour $c(e) \in \{u, v\}$, so that it is monochromatic in colour x^e_r: first play a single move to make all of G'_e

except for a single island monochromatic, then play colours $x_1^e, \ldots, x_r^e$ in this central component. Ignoring the islands for the moment, the components corresponding to each G_e now have distinct colours, so we can link these components with a minimum of $m-1$ moves. Finally, we need to flood the islands, and this requires exactly $|\{c(e) : e \in E\}|$ moves. But we know that G has a vertex cover of size at most k, say V'. By the definition of a vertex cover, if the gadget G'_e uses colours u and v, then at least one of $u, v \in V'$. So for each G_e, we may choose to leave an island of colour d where $d \in V'$. Following this strategy, we are left in the final stage with islands of at most k distinct colours, and can flood these in k steps (by cycling through each colour in turn in the external monochromatic component). Hence we can flood B_G in $N + k$ steps. □

Lemma 4.3. *If we can flood B_G in $N + k$ steps (for some $0 \leq k \leq |V|$), then G has a vertex cover of size at most k.*

Proof. Suppose the sequence S floods B_G, where $|S| = N+k$. Observe that, if we contract monochromatic components of the coloured graph corresponding to B_G, we obtain a tree T. Let P be the unique path in P joining the two vertices in T that correspond to the monochromatic components incident with opposite ends of the board and note that, by Lemma 3.8, we may assume that all moves of S are played in trunk$(T) \subseteq P$. Moreover, S must flood P when played in this isolated path.

We will say that a component of colour d is *eliminated* by the move α if α changes the colour of that component, linking it to an adjacent component of colour $d' \neq d$. We say that α *eliminates the colour* d if it removes the last component of colour d remaining in the graph.

Suppose that, for some $v \in V$, a single move $\alpha \in S$ eliminates two components A_1 and A_2 of P that both have colour v initially. A_1 and A_2 cannot belong to the same gadget G_e so, for α to eliminate them both, the moves played in S before α must create a single monochromatic component A containing both A_1 and A_2. Such a component A must contain $i \geq 2r$ different colours under the original colouring, and so S must include at least $i-1$ moves played in this section of the path before α (so in total at least i moves of S are played in A). But there are at least $mr - (i - 2r)$ colours on the path outside A (as at least $2r$ of the colours in A must also appear outside A), and all but one of these must be eliminated by moves played outside A. This gives

$$|S| \geq mr + 2r - 1 > N + |V|,$$

a contradiction. So we may assume that no move in S eliminates more than one component that originally has colour $v \in V$.

Now consider the leaves of T, and let $\bar{S}$ be the set of moves in S that eliminate the second leaf in each G'_e. Suppose that one leaf in G'_e has already been eliminated, and that the move $\alpha \in \bar{S}$ removes the second leaf. Since one leaf has already been eliminated, no components in $G_e \cap P$ which originally had a colour $v \in V$ still have colour v. Suppose that α reduces the number of monochromatic components on P. By the reasoning above, if α links G'_e to another component outside G_e that originally had colour v, we would have a contradiction with $|S| > N + |V|$, so in fact α must link G'_e to a component in G_e whose colour was previously changed to v by some move β; such a move β could not decrease the number of monochromatic components of P. Thus, for every $\alpha \in \bar{S}$, there is at least one move of S that does not decrease the number of monochromatic components of P.

Hence we see that

$$|S| \geq mr + 2m - 1 + |\bar{S}| = N + |\bar{S}|,$$

and so $|\bar{S}| \leq k$. However, we know that $\bar{S}$ eliminates at least one leaf from every G'_e, and clearly each move in $\bar{S}$ can eliminate leaves of only one colour. Hence there exists some set $C' \subset V$ such that $|C'| \leq |\bar{S}| \leq k$ and at least one leaf in each G'_e has a colour from C'. In other words, $|C'| \leq k$, and for every edge $uv \in E$, $\{u, v\} \cap C' \neq \emptyset$, so C' is in fact a vertex cover for G of size at most k. □

Proof of Theorem 4.1. The reduction from Vertex Cover is immediate from Lemmas 4.2 and 4.3. □

5 Conclusions and open problems

We have demonstrated an algorithm which shows that the problem c-Free-Flood-It, restricted to $2 \times n$ boards, is fixed parameter tractable with parameter c, and on the other hand we have shown that Free-Flood-It remains NP-hard in this setting. This answers an open question from [7], in which Clifford, Jalsenius, Montanaro and Sach showed that Fixed-Flood-It can be solved in time $O(n)$ on such boards. Our results therefore give the first example of a class of graphs on which the complexity status of the fixed and free versions of the game differ.

Together with results from [7] and [14], this almost completes the picture for the complexity of flood-filling problems restricted to $k \times n$ boards. However, there does remain one open case:

Problem 1. *What are the complexities of 3-*Fixed-Flood-It *and 3-*Free-Flood-It *restricted to $k \times n$ boards, in the case that $k \geq 3$ is a fixed integer?*

Another interesting direction for further research would be to consider extremal flood-filling problems in this setting.

Problem 2. *What colourings of a $k \times n$ board B with c colours give the maximum value of $m(B)$?*

As a first step, it should not be hard to determine the maximum value of $m(B)$ for a $1 \times n$ board.

Such questions can also be generalised to arbitrary graphs, leading to two more natural questions.

Problem 3. *Given a graph G and an integer $c \geq \chi(G)$, what proper colourings ω of G with exactly c colours maximise $m(G, \omega)$?*

Problem 4. *Given a graph G, what proper colourings ω minimise $m(G, \omega)$? Do such colourings necessarily use exactly $\chi(G)$ colours?*

References

[1] *Flood It Game*, http://floodit.appspot.com.

[2] *Flood It! 2*, available at http://itunes.apple.com.

[3] *Flood It!*, available at https://market.android.com.

[4] *Mad Virus*, http://www.bubblebox.com/play/puzzle/539.htm.

[5] David Arthur, Raphaël Clifford, Markus Jalsenius, Ashley Montanaro, and Benjamin Sach, The Complexity of Flood Filling Games, in Paolo Boldi and Luisa Gargano, editors, *FUN*, volume 6099 of *Lecture Notes in Computer Science*, Springer, ISBN 978-3-642-13121-9, 2010, pages 307-318.

[6] A. Born, Flash application for the computer game *Biene (Honey-Bee)*, 2009. http://www.ursulinen.asn-graz.ac.at/Bugs/htm/games/biene.htm.

[7] Raphaël Clifford, Markus Jalsenius, Ashley Montanaro, and Benjamin Sach, The Complexity of Flood Filling Games, *Theory of Computing Systems* **50** (2012), pp. 72-92.

[8] Thomas H. Cormen, Charles E. Leiserson and Ronald L. Rivest, *Introduction to Algorithms*, MIT Press and McGraw-Hill, 1990.

[9] Rudolf Fleischer and Gerard J. Woeginger, An Algorithmic Analysis of the Honey-Bee Game, *Theoretical Computer Science* **452** (2012), pp. 75-87.

[10] H. Fukui, A. Nakanishi, R. Uehara, T. Uno, Y. Uno, The complexity of free flooding games, Information Processing Society of Japan (IPSJ) SIG Notes 2011 (August 2011), 1-5.

[11] R. M. Karp, Reducibility among combinatorial problems, in R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, Plenum Press, New York, 1972.

[12] Aurélie Lagoutte, Jeux d'inondation dans les graphes, Technical report, ENS Lyon, HAL: hal-00509488, August 2010.

[13] A. Lagoutte, M. Noual, E. Thierry, Flooding games on graphs, HAL: hal-00653714, December 2011.

[14] Kitty Meeks and Alexander Scott, The complexity of flood-filling games on graphs, *Discrete Applied Mathematics* **160** (2012), pp. 959-969.

[15] Kitty Meeks and Alexander Scott, Spanning trees and the complexity of flood-filling games, in Kranakis, Krizanc and Luccio, editors, *FUN*, volume 7288 of *Lecture Notes in Computer Science*, Springer, ISBN 978-3-642-30346-3, 2012, pages 282-292.

[16] Kitty Meeks and Alexander Scott, Spanning trees and the complexity of flood-filling games, arXiv:1203.2538v2 [cs.DS], 2012.